安裝Node.js,npm,Socket.IO,Vue.js,yarn,sqlite,mangodb步驟：

安裝Node.js:$sudo apt-get install Node.js

安裝npm:$sudo apt-get install npm

$ npm init

$ npm install express --save

安裝Socket.IO:(用npm):$sudo npm install -gn

$npm install Socket.IO

安裝Vue.js:$sudo apt-get install Vue.js

安裝yarn:$sudo apt-get update&&sudo apt-get install yarn

安裝sqlite:$sudo apt-get install sqlite

安裝mangodb:$sudo apt-get update

$sudo apt-get install mongodb

看node.js版本:$Node -v

看code:$sudo snap install code

使用yarn:$npm install

$yarn

初始包的開發環境:$npm install

$yarn

安裝依賴到package中:$npm uninstall -save

$yarn hemove

安裝package文件定義的所有依賴:$npm install --save

$yarn add

移除依賴項目:$npm install -save dev

$yarn add dev

安裝開發依賴項目:$pm update -save

$yarn upgrade

更新依賴項目:$npm install --global

$yarn global add

安裝全局依賴項目:$npm run/test

$yarn run/test

npm指令：

npm -v npm版本控制

npm list 列出目前工作目錄安裝的套件

npm install module \_name 安裝於目前目錄

npm install -g module \_name 安裝於系統目錄

npm update module \_name 更新套件

Vue 常用指令：

* v-text
* v-html
* v-show
* v-if
* v-else
* v-else-if
* v-for
* v-on
* v-bind
* v-model
* v-pre
* v-cloak
* v-once

## **Vue指令说明**

|  |  |
| --- | --- |
| **指令** | **说明** |
| [**v-text**](https://haicoder.net/vue/vue-vtext.html) | vue v-text 指令主要用来更新 textContent，可以等同于 JS 的 text 属性。 |
| [**v-html**](https://haicoder.net/vue/vue-vhtml.html) | 双大括号的方式会将数据解释为纯文本，而非 HTML。为了输出真正的 HTML，可以用 vue v-html 指令。它等同于 JS 的 innerHtml 属性。 |
| [**v-pre**](https://haicoder.net/vue/vue-vpre.html) | vue v-pre 指令主要用来跳过这个元素和它的子元素编译过程。可以用来显示原始的 Mustache 标签。跳过大量没有指令的节点加快编译。 |
| [**v-cloak**](https://haicoder.net/vue/vue-vcloak.html) | vue v-cloak 指令是用来保持在元素上直到关联实例结束时进行编译。 |
| [**v-once**](https://haicoder.net/vue/vue-vonce.html) | vue v-once 指令关联的实例，只会渲染一次。之后的重新渲染，实例极其所有的子节点将被视为静态内容跳过，这可以用于优化更新性能。 |
| [**v-if**](https://haicoder.net/vue/vue-vif.html) | vue v-if 指令可以实现条件渲染，Vue 会根据表达式的值的真假条件来渲染元素。 |
| [**v-else**](https://haicoder.net/vue/vue-velse.html) | vue v-else 指令是搭配 v-if 使用的，它必须紧跟在 v-if 或者 v-else-if 后面，否则不起作用。 |
| [**v-else-if**](https://haicoder.net/vue/vue-velse-if.html) | vue v-else-if 指令充当 v-if 的 else-if 块，可以链式的使用多次。可以更加方便的实现 switch 语句。 |
| [**v-show**](https://haicoder.net/vue/vue-vshow.html) | vue v-show 指令也是用于根据条件展示元素。 |
| [**v-for**](https://haicoder.net/vue/vue-vfor.html) | vue v-for 指令根据遍历数组来进行渲染。 |
| [**v-bind**](https://haicoder.net/vue/vue-vbind.html) | vue v-bind 指令用来动态的绑定一个或者多个特性。没有参数时，可以绑定到一个包含键值对的对象。常用于动态绑定 class 和 style，以及 href 等。 |
| [**v-model**](https://haicoder.net/vue/vue-vmodel.html) | vue v-model 指令用于在表单上创建双向数据绑定。 |
| [**v-on**](https://haicoder.net/vue/vue-von.html) | vue v-on 指令主要用来监听 dom 事件，以便执行一些代码块。表达式可以是一个方法名。 |

**建立資料庫檔案**

**用sqlite3建立資料庫的方法很簡單，只要在shell下鍵入（以下$符號為shell提示號，請勿鍵入）：**

**$ sqlite3 foo.db3**

**如果目錄下沒有foo.db3，sqlite3就會建立這個資料庫。sqlite3並沒有強制資料庫檔名要怎麼取，因此如果你喜歡，也可以取個例如foo.icannameitwhateverilike的檔名。**

**在sqlite3提示列下操作**

**kl;kjlkjljklkjkj 進入了sqlite3之後，會看到以下文字：**

**SQLite version 3.1.3**

**Enter ".help" for instructions**

**sqlite＞**

**這時如果使用.help可 以取得求助，.quit則是離開（請注意：不是quit）**

**SQL的指令格式**

**所有的SQL指令都是以分號（;）結尾的。如果遇到兩個減號（--）則代表註解，sqlite3會略過去。**

**建立資料表**

**假設我們要建一個名叫film的資料表，只要鍵入以下指令就可以了：**

**create table film(title, length, year, starring);**

**這樣我們就建立了一個名叫film的資料表，裡面有name、length、year、starring四個欄位。**

**這個create table指令的語法為：**

**create table table\_name(field1, field2, field3, ...);**

**table\_name是資料表的名稱，fieldx則是欄位的名字。sqlite3與許多SQL資料庫軟體不同的是，它不在乎欄位屬於哪一種資料型態：sqlite3的欄位可以儲存任何東西：文字、數字、大量文字（blob），它會在適時自動轉換。**

**建立索引**

**如果資料表有相當多的資料，我們便會建立索引來加快速度。好比說：**

**create index film\_title\_index on film(title);**

**意思是針對film資料表的title欄位，建立一個名叫film\_title\_index的索引。這個指令的語法為**

**create index index\_name on table\_name(field\_to\_be\_indexed);**

**一旦建立了索引，sqlite3會在針對該欄位作查詢時，自動使用該索引。這一切的操作都是在幕後自動發生的，無須使用者特別指令。**

**加入一筆資料**

**接下來我們要加入資料了，加入的方法為使用insert into指令，語法為：**

**insert into table\_name values(data1, data2, data3, ...);**

**例如我們可以加入**

**insert into film values ('Silence of the Lambs, The', 118, 1991, 'Jodie Foster');**

**insert into film values ('Contact', 153, 1997, 'Jodie Foster');**

**insert into film values ('Crouching Tiger, Hidden Dragon', 120, 2000, 'Yun-Fat Chow');**

**insert into film values ('Hours, The', 114, 2002, 'Nicole Kidman');**

**如果該欄位沒有資料，我們可以填NULL。**

**查詢資料**

**講到這裡，我們終於要開始介紹SQL最強大的select指令了。我們首先簡單介紹select的基本句型：**

**select columns from table\_name where expression;**

**最常見的用法，當然是倒出所有資料庫的內容：**

**select \* from film;**

**如果資料太多了，我們或許會想限制筆數：**

**select \* from film limit 10;**

**或是年份比較早的電影先列出來（預設為 asc）：**

**select \* from film order by year limit 10;**

**或是年份比較晚的電影先列出來：**

**select \* from film order by year desc limit 10;**

**或是我們只想看電影名稱跟年份：**

**select title, year from film order by year desc limit 10;**

**查所有茱蒂佛斯特演過的電影：**

**select \* from film where starring='Jodie Foster';**

**查所有演員名字開頭叫茱蒂的電影（'%'、'\_' 符號便是 SQL 的萬用字元，前者代表任意長度字元，後者代表任意一個字元）：**

**select \* from film where starring like 'Jodie%';**

**查所有演員名字以茱蒂開頭、年份晚於1985年、年份晚的優先列出、最多十筆，只列出電影名稱和年份：**

**select title, year from film where starring like 'Jodie%' and year ＞= 1985 order by year desc limit 10;**

**有時候我們只想知道資料庫一共有多少筆資料：**

**select count(\*) from film;**

**有時候我們只想知道1985年以後的電影有幾部：**

**select count(\*) from film where year >= 1985;**

**（進一步的各種組合，要去看SQL專書，不過你大概已經知道SQL為什麼這麼流行了：這種語言允許你將各種查詢條件組合在一起──而我們還沒提到「跨資料庫的聯合查詢」呢！）**

**如何更改或刪除資料**

**瞭解select的用法非常重要，因為要在sqlite更改或刪除一筆資料，也是靠同樣的語法。**

**例如有一筆資料的名字打錯了：**

**update film set starring='Jodie Foster' where starring='Jodee Foster';**

**就會把主角欄位裡，被打成'Jodee Foster'的那筆（或多筆）資料，改回成Jodie Foster。**

**delete from film where year ＞ 1970;**

**sqlite可以在shell底下直接執行命令：**

**sqlite3 film.db "select \* from film;"**

**輸出 HTML 表格：**

**sqlite3 -html film.db "select \* from film;"**

**將資料庫「倒出來」：**

**sqlite3 film.db ".dump" ＞ output.sql**

**利用輸出的資料，建立一個一模一樣的資料庫（加上以上指令，就是標準的SQL資料庫備份了）：**

**sqlite3 film.db ＜ output.sql**

**在大量插入資料時，你可能會需要先打這個指令：**

**begin;**

**插入完資料後要記得打這個指令，資料才會寫進資料庫中：**

**commit;**

**sqlite特點：文件型數據庫:**常用於應用程序存儲數據，比如瀏覽器存儲用戶的資料等GUI

**mangodb特點:NoSQL數據庫**：不使用SQL語句進行檢索MongoDB存儲文件的類型**類似於JSON**格式

**MongoDB:**Mongo是所有NoSQL數據庫中最廣為人知的，並且是JS密集型MEAN堆棧的組成部分。它在企業運營中很受歡迎，尤其是那些對數據有極高要求的企業。Facebook，谷歌，Adobe，Squarespace甚至匯豐銀行和花旗銀行等銀行都在使用它。首先，它是免費的並且是開源的，因此，如果您在柵欄上，那麼將腳趾浸入的成本是最小的。其次，在處理大量低複雜度數據時它非常快。Mongo的理想用例就像氣象站一樣：它每秒收集多個不同的數據類型，但是數據本身非常簡單。MongoDB是無模式的，這意味著您可以根據自己的需要製作任何喜歡的模式。Mongoose ODM可以使此過程變得更容易，它可以幫助您對應用程序數據進行建模，包括內置的驗證，類型轉換，查詢構建以及總體上需要編寫的代碼量。這種結構（與內置分片結合）確實允許高度的水平可伸縮性，並使Mongo非常靈活。最重要的是，類似於JSON的工件存儲使其可以存儲比傳統RDBMS更為廣泛的數據類型，使其更適合需要存儲更多異常數據類型的數據庫。**SQLite:**完全相反，SQLite是一種輕量級的無服務器RDBMS，在設計時考慮到了經濟性和效率，最適合於像單個應用程序或IoT設備之類的小規模運營。許多移動應用程序都運行SQLite，甚至Android操作系統也使用它。Windows 10的幾個核心組件以及RedHat（RPM）程序包管理器也使用它-除了RedHat Linux之外，RedHat Linux還被一些流行的Linux發行版（例如CentOS和Fedora）使用。SQLite支持同時具有多個數據庫用戶，但是它為讀取和寫入操作鎖定了整個數據庫，使其不適用於大量並發的讀取/寫入操作。儘管每次操作的延遲為毫秒，但是大量的用戶使這成為一個問題。如果您來自MySQL或其他SQL DB，則適應SQLite的時間會容易得多：幾乎沒有學習曲線。它也是無服務器的，可以進一步減少混亂—它直接進入磁盤上的數據庫。SQLite不需要管理員，幾乎不需要維護。這使得它非常適合於家用電子產品和其他用戶難以致電求助的系統。

|  |  |  |
| --- | --- | --- |
| **SQLite數據類型** |  | 使用SQL可分為以下四個動作 |
|  |  |  |
| **一般數據采用的固定的靜態數據類型，而SQLite采用的是動態數據類型，會根據存入值自動判斷。SQLite具有以下五種常用的數據類型：** |  | 1.新增記錄（Insert）。 |
| **NULL: 這個值為空值;** |  | 2.查詢記錄（Query）。 |
| **VARCHAR(n)：長度不固定且其最大長度為 n 的字串，n不能超過 4000;** |  | 3.更改記錄（Update）。 |
| **CHAR(n)：長度固定為n的字串，n不能超過 254;** |  | 4.刪除記錄（Delete）。 |
| **INTEGER: 值被標識為整數,依據值的大小可以依次被存儲為1,2,3,4,5,6,7,8;** |  |  |
| **REAL: 所有值都是浮動的數值,被存儲為8字節的IEEE浮動標記序號;** |  | 以下先來決定資料表的欄位,如果範例為多人聊天紀錄,就必須有以下欄位: |
| **TEXT: 值為文本字符串,使用數據庫編碼存儲(TUTF-8, UTF-16BE or UTF-16-LE);** |  |  |
| **BLOB: 值是BLOB數據塊，以輸入的數據格式進行存儲。如何輸入就如何存儲,不改 變格式;** |  |  |
| **DATA ：包含了 年份、月份、日期;** |  | 1.用來判斷第幾筆資料的主鍵的:\_id |
| **TIME： 包含了 小時、分鍾、秒;** |  | 2.聊天紀錄說話者姓名:name |
| **SQLite常用的方法** |  | 3.說的內容:content |
| **方法名稱** | **方法表示含義** | 4.字的顏色:color |
| **openOrCreateDatabase(String path,SQLiteDatabase.CursorFactory factory) 創建數據庫** | **打開或創建數據庫** | 5.多人聊天的玩家號碼:number |
| **insert(String table,String nullColumnHack,ContentValues values)插入數據** | **插入一條記錄** |  |
| **delete(String table,String whereClause,String[] whereArgs)刪除數據** | **刪除一條記錄** |  |
| **query(String table,String[] columns,String selection,String[] selectionArgs,String groupBy,String having,String orderBy) 查找數據** | **查詢一條記錄** |  |
| **update(String table,ContentValues values,String whereClause,String[] whereArgs) 更新數據** | **修改記錄** |  |
| **execSQL(String sql)** | **執行一條SQL語句** |  |
| **close() 關閉數據** |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 關聯式資料庫 | NoSQL 資料庫 |  |  |  |  |  |
| **最佳工作負載** | 關聯式資料庫專門用於交易性以及高度一致性的線上交易處理 (OLTP) 應用程式，並且非常適合於線上分析處理 (OLAP) 使用。 | NoSQL 資料庫專門用於包含低延遲應用程式的多樣資料存取模式。NoSQL 搜尋資料庫專門用於進行半結構資料的分析。 |  |  |  |  |  |
| **資料模型** | 關聯式模型將資料標準化，成為由列和欄組成的表格。結構描述嚴格定義表格、列、欄、索引、表格之間的關係，以及其他資料庫元素。此類資料庫強化資料庫表格間的參考完整性。 | NoSQL 資料庫提供鍵值、文件和圖形等多種資料模型，具有最佳化的效能與規模。 |  |  |  |  |  |
| **ACID 屬性** | 關聯式資料庫則提供單元性、一致性、隔離性和耐用性 (ACID) 的屬性：  單元性要求交易完整執行或完全不執行。  一致性要求進行交易時資料就必須符合資料庫結構描述。  隔離性要求並行的交易必須分開執行。  耐用性要求從意外的系統故障或停電狀況還原成上個已知狀態的能力。 | NoSQL 資料庫通常透過鬆綁部分關聯式資料庫的 ACID 屬性來取捨，以達到能夠橫向擴展的更彈性化資料模型。這使得 NoSQL 資料庫成為橫向擴展超過單執行個體上限的高吞吐量、低延遲使用案例的最佳選擇。 |  |  |  |  |  |
| **效能** | 一般而言，效能取決於磁碟子系統。若要達到頂級效能，通常必須針對查詢、索引及表格結構進行優化。 | 效能通常會受到基礎硬體叢集大小、網路延遲，以及呼叫應用程式的影響。 |  |  |  |  |  |
| **擴展** | 關聯式資料庫通常透過增加硬體運算能力向上擴展，或以新增唯讀工作負載複本的方式向外擴展。 | NoSQL 資料庫通常可分割，因為存取模式可透過使用分散式架構來向外擴展，以近乎無限規模的方式提供一致效能來增加資料吞吐量。 |  |  |  |  |  |
| **API** | 存放和擷取資料的請求是透過符合結構式查詢語言 (SQL) 的查詢進行通訊。這些查詢是由關聯式資料庫剖析和執行。 | 以物件為基礎的 API 讓應用程式開發人員可輕鬆存放和擷取資料結構。應用程式可透過分區索引鍵查詢鍵值組、欄集，或包含序列化應用程式物件與屬性的半結構化文件。 |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| 名稱 | MongoDB X | SQLite X |
| 描述 | 最受歡迎的文檔存儲之一，既可以作為完全託管的雲服務使用，也可以在自助管理的基礎架構上進行部署 | 廣泛使用的過程RDBMS |
| 主數據庫模型 | [文件存儲](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://db-engines.com/en/article/Document%2BStores&usg=ALkJrhhO3FaUf0Z3xkTpnNIDpua7pLj2uQ) | [關係數據庫管理系統](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://db-engines.com/en/article/RDBMS&usg=ALkJrhiq-MLURlufE3pxDh_pHGClepB8MQ) |
| 二級數據庫模型 | [搜索引擎](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://db-engines.com/en/article/Search%2BEngines&usg=ALkJrhgYKs9gSjg-ifyOTT4XkDukcOEHnw) |  |
|  |  |  |
| 網站 | [www.mongodb.com](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://www.mongodb.com/&usg=ALkJrhj6o_zTSmHg0A_oqU7OHWbQtXdepg) | [www.sqlite.org](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://www.sqlite.org/&usg=ALkJrhiy4UXzbBXYJvQTO3yK9J_nWnpy6A) |
| 技術文檔 | [docs.mongodb.com/manual](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://docs.mongodb.com/manual/&usg=ALkJrhjWmI2SbhSnLhkeLwEUZ6XOsajyJw) | [www.sqlite.org/docs.html](https://translate.googleusercontent.com/translate_c?depth=1&hl=zh-TW&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=https://www.sqlite.org/docs.html&usg=ALkJrhiP_B7VUeGSUy578QGNLaFCnEkn6Q) |
| 開發者 | MongoDB，Inc | 德韋恩·理查德·希普 |
| 初始發行 | 2009年 | 2000 |
| 當前版本 | 2020年6月4.2.8 | 3.32.2（2020年6月4日），2020年6月 |
| 執照 | 開源的 | 開源的 |
| 僅基於雲 | 沒有 | 沒有 |
| DBaaS產品（贊助商鏈接） | ScaleGrid for MongoDB：在AWS，Azure和DigitalOcean上對MongoDB數據庫進行完全託管的託管，在​​＃1多雲DBaaS上具有高可用性和SSH訪問權限。  MongoDB Atlas：按需部署完全託管的雲數據庫，並在幾分鐘內即可使用。在AWS，Azure和GCP上可用。 |  |
| 實施語言 | C ++ | C |
| 服務器操作系統 | Linux  OS X  Solaris  Windows | 無服務器 |
| 數據方案 | 無模式 | 是 |
| 打字 | 是 | 是 |
| XML支持 |  | 沒有 |
| 次要指標 | 是 | 是 |
| 的SQL | 通過MongoDB Connector for BI的只讀SQL查詢 | 是 |
| API和其他訪問方法 | 使用JSON的專有協議 | ADO.NET JDBC ODBC |
| 支持的編程語言 | 動作腳本C C＃C ++ Clojure ColdFusion D Dart Delphi Erlang Go Groovy Haskell Java JavaScript Lisp Lua MatLab Perl PHP PowerShell Prolog Python R Ruby Scala Smalltalk | Actionscript  Ada  Basic  C  C＃  C ++  D  Delphi  Forth  Fortran  Haskell  Java  JavaScript  Lisp  Lua  MatLab  Objective-C  OCaml  Perl  PHP  PL / SQL  Python  R  Ruby  Scala  Scheme  Smalltalk  Tcl |
| 服務器端腳本 | 的JavaScript | 沒有 |
| 扳機 | 是 | 是 |
| 分割方法 | 分片 | 沒有 |
| 複製方式 | 源副本複制 | 沒有 |
| MapReduce | 是 | 沒有 |
| 一致性概念 | 最終一致性  立即一致性 |  |
| 外鍵 | 沒有 | 是 |
| 交易概念 | 具有快照隔離功能的多文檔ACID事務 | 酸 |
| 並發 | 是 | 是 |
| 耐用性 | 是 | 是 |
| 內存功能 | 是 | 是 |
| 用戶概念 | 用戶和角色的訪問權限 | 沒有 |

SQLite不支援加密,所以我們針對資料庫檔案加密。現有兩種解決方案:

**i>對資料庫中的資料進行加密**

①優點:

加密速度快。

程式無須變動,只針對資料進行加密。

②缺點:

來回加密,造成處理資料緩慢。

僅對資料加密,依然可以看到資料表的sql語句,可能猜測到表的作用。

③實現:

一是:對明文資料進行加密返回密文資料

二是:對密文資料進行解密返回明文資料

**ii>對資料庫檔案進行加密**

①優點

對整個檔案進行了加密,使用者通過編輯器看不到任何有用的資料。

進行資料庫開啟時,使用程式即可解密檔案。

②缺點

需要修改sqlite原始碼,這個工作難度比較大。

③實現(太難,本人並沒有進行實驗)

a.修改sqlite原始碼,追加對資料庫檔案進行加密的功能。

b.編譯含有加密功能的程式原始碼,生成各自平臺需要使用的庫檔案。

c.將加密sqlite庫檔案引入各自平臺中,修改資料庫訪問層程式碼。

d.進行程式的部署,測試。

# **sqlite3的特殊命令**

大多數候，sqlite3讀入輸入行，並把它們傳遞到SQLite庫中去運行。但是如果輸入行以一個點（“.”）開始，那麼這行將被sqlite3程序自 己截取並解釋。這些“點命令”通常被用來改變查詢輸出的格式，或者執行鞭個預封包（預定義prepackaged）的查詢語句。

你可以在任何時候輸入“.help”，列出可用的點命令。例如

sqlite> .help

## **.bail ON|OFF**

Stop after hitting an error. Default OFF(遇到錯誤時不再繼續, 默認爲OFF)

## **.databases**

List names and files of attached databases(列出附加到數據庫的數據庫和文件)

## **.dump ?TABLE? ...**

Dump the database in an SQL text format(保存表到SQL格式的文件中, 沒有指表名, 則保存所有. 如果要保存到磁盤上需要結合 .output 命令.)

## **.echo ON|OFF**

Turn command echo on or off(打開/關閉 命令行回顯)

## **.exit**

Exit this program(退出該命令行)

## **.explain ON|OFF**

Turn output mode suitable for EXPLAIN on or off.( 以合適的方式顯示錶頭, 不帶參數則爲開啓)

## **.header(s) ON|OFF**

Turn display of headers on or off(是否顯示錶頭, 和 .explain 差別不是很大)

## **.help**

Show this message(顯示幫助信息)

## **.import FILE TABLE**

Import data from FILE into TABLE(從文件中導入表)

## **.indices TABLE**

Show names of all indices on TABLE(顯示索引)

## **.load FILE ?ENTRY?**

Load an extension library(加載一個擴展庫)

## **.mode MODE ?TABLE?**

Set output mode where MODE is one of:

(設置輸出模式, 模式可以是以下幾種):

csv Comma-separated values( 以逗號分隔的值)

column Left-aligned columns. (See .width)( 表頭左對齊(參見 .width))

html HTML <table> code(顯示 HTML 代碼)

insert SQL insert statements for TABLE( SQL插入語句)

line One value per line( 一行一個值)

list Values delimited by separator string(值用 string 分隔)

tabs Tab-separated values(以 tab 分隔的值)

tcl TCL list elements(TCL 列表元素)

## **.nullvalue STRING**

Print STRING in place of NULL values(以 STRING 代替 NULL 值的輸出)

## **.output FILENAME**

Send output to FILENAME(輸出到文件, 而不是顯示在屏幕上)

## **.output stdout**

Send output to the screen(輸出到屏幕上)

## **.prompt MAIN CONTINUE**

Replace the standard prompts(替換默認的命令提示信息, 默認就是 sqlite>)

## **.quit**

Exit this program(退出命令行)

## **.read FILENAME**

Execute SQL in FILENAME(執行 FILENAME 中的 SQL語句)

## **.schema ?TABLE?**

Show the CREATE statements(顯示 CREATE 語句)

## **.separator STRING**

Change separator used by output mode and .import

## **.show**

Show the current values for various settings(顯示各種設置)

## **.tables ?PATTERN?**

List names of tables matching a LIKE pattern(查看數據庫的表列表)

## **.timeout MS**

Try opening locked tables for MS milliseconds(在 MS 時間內嘗試打開被鎖定的表)

## **.width NUM NUM ...**

Set column widths for "column" mode(設置 column 模式中的列的寬度)

## **.timer ON|OFF**

(顯示CPU時間)

## **.vfsname ?AUX?**

(顯示 VFS 棧信息)

## **.restore ?DB? FILE**

(從文件中還原數據到表, 默認表爲 main)

sqlite>